NISTIR 5460

Adaptive, Predictive 2-D Feature Tracking Algorithm for Finding the Focus of Expansion

Srivatsan Krishnan
Florida Atlantic University
The Robotics Center and
The Electrical Engineering Department
Boca Raton, Florida

and

Daniel Raviv
Florida Atlantic University
The Robotics Center and
The Electrical Engineering Department
Boca Raton, Florida
and
Sensory Intelligent Group
Intelligent Systems Division

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Bldg. 220 Rm. B124
Gaithersburg, MD 20899

NIST

NISTIR 5460

Adaptive, Predictive 2-D Feature Tracking Algorithm for Finding the Focus of Expansion

Srivatsan Krishnan
Florida Atlantic University
The Robotics Center and
The Electrical Engineering Department
Boca Raton, Florida

and

Daniel Raviv
Florida Atlantic University
The Robotics Center and
The Electrical Engineering Department
Boca Raton, Florida
and
Sensory Intelligent Group
Intelligent Systems Division

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Bldg. 220 Rm. B124
Gaithersburg, MD 20899

June 1994

U.S. DEPARTMENT OF COMMERCE
Ronald H. Brown, Secretary

TECHNOLOGY ADMINISTRATION
Mary L. Good, Under Secretary for Technology

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Arati Prabhakar, Director

ADAPTIVE, PREDICTIVE 2-D FEATURE TRACKING ALGORITHM FOR FINDING THE FOCUS OF EXPANSION *

Srivatsan Krishnan[1] and Daniel Raviv[1,2]

email : s_krishn@acc.fau.edu ; ravivd@acc.fau.edu

[1]Robotics Center and Department of Electrical Engineering,
Florida Atlantic University, Boca Raton, Fl 33431.

[2]Intelligent Systems Division
National Institute of Standards and Technology (NIST)
Bldg.220, Room B124, Gaithersburg, MD 20899.

Abstract

This paper describes a robust, pixel-based adaptive algorithm to track image features, both spatially and temporally, over a sequence of monocular images. The algorithm assumes no *a priori* knowledge about the features to be tracked, or the motion of the camera or the objects, in the 3-D scene. The features to be tracked are selected by the algorithm and these correspond to the peaks of a *'matching surface'* constructed from the first image of the sequence to be analyzed. *Any kind of motion* can be tolerated keeping in mind the pixels- per-frame motion limitations. In our experiments, we have used the algorithm for tracking up to 3 pixels motion between frames with absolutely *no subpixel calculations* being necessary. Taking into account constraints of temporal continuity, the algorithm uses simple and efficient predictive tracking over multiple frames. The algorithm accepts a slow, continuous change of brightness D.C. level in the pixels of the feature. Trajectories of features on multiple objects can be computed. As an application of the algorithm we show how the algorithm can be used to find the Focus of Expansion (FOE) using a sequence of real images.

Index Terms- Matching, feature tracking, visual motion

---

* This work was supported in part by a grant to Florida Atlantic University from the National Science Foundation, Division of Information, Robotics and Intelligent Systems, Grant # IRI-9115939

1. INTRODUCTION

It is well known that a significant amount of useful information can be obtained about 3-D motion and structure using the projected 2-D image motion sequence resulting from the movement of a camera and/or that of 3-D objects. The various approaches to motion analysis can be broadly classified into those using optical flow techniques and those using feature tracking techniques.

Optical flow techniques are extremely useful in the analysis of motion by a monocular observer. However, they are in general noisy and inaccurate. Moreover, most methods need pre-processing of the image before optical flow techniques can be applied. A good reference for motion analysis using optical flow techniques can be found in [17].

A number of researchers have developed 2-D feature tracking algorithms for the analysis of motion. Until recently, many of these were based on the assumption that *a priori* knowledge about the camera motion and/or, the 3-D environment or its 2-D projection, was available. Methods which do not need prior knowledge about the motion have the advantage of applicability to a variety of situations.

This paper describes a feature tracking algorithm that needs no *a priori* knowledge about either the motion of the camera or the objects in the 3-D scene or their 2-D projection in the image.

1.1 PREVIOUS WORK

In this section, we give an overview of some of the methods for feature tracking and the extraction of the Focus of Expansion found in the literature.

Various approaches to motion analysis developed till 1989 can be found in [28].

Figure 1.1 is a diagram that gives a classification of some of the feature tracking methods in the literature.

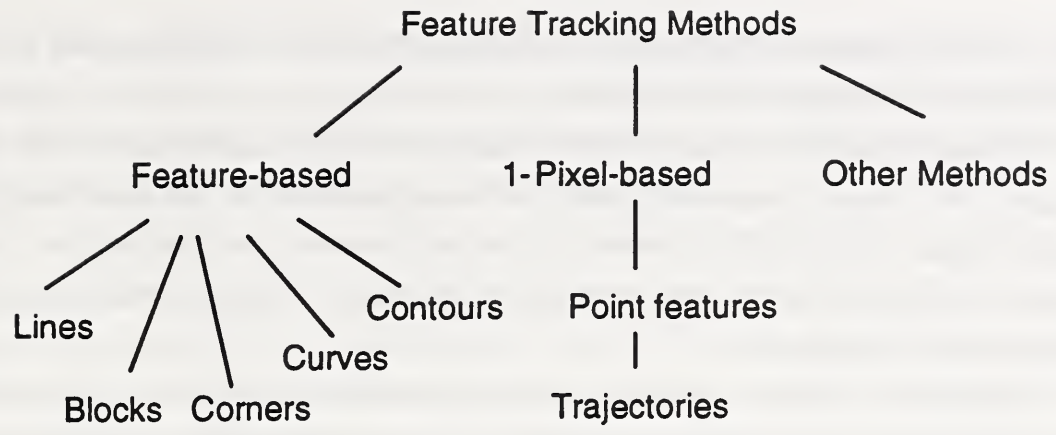


Figure 1.1 : A diagram giving a classification of feature tracking methods.

1.1.1 Feature Tracking Methods - An Overview

Tracking *line segments* is one approach to solve the feature tracking problem as presented in [30] and [6]. Using straight line correspondences, one can determine the motion parameters as well as structure information of the objects. Line tracking may fail in situations which give rise to the so-called 'aperture problem'.

Block matching is another approach to tackle the feature tracking problem [11], [31], [22], [16] which is based on tracking coherent entities together. As described in [11], the image is divided into fixed size areas and the best match is found by minimizing a distortion function between the reference image and the desired image. Motion estimation can be done using block matching. In [22], the velocity vector approach is used, where elements of a region moving coherently share a common velocity vector. All these algorithms are based on the coherence in a region. The tradeoff in the block matching approach would be in determining the size of the blocks for optimum speed of computations as well as the size necessary for accuracy.

The 3-D model has been used in quite a few approaches to motion analysis [8], [9], [3]. Gennery, [8], was among the earlier attempts at tracking known 3-D objects. The motion parameters are used in the analysis of motion. Prediction is usually done using Kalman Filtering or its modifications. In practical situations, the 3-D objects to be tracked, or the motion of the camera, may not always be known .

Curves and *contours* can also be used for tracking purposes [2],[19]. An algorithm based on energy minimization is presented in [2], the attempt being to preserve the matching of high curvature, at the same time ensuring a smooth field of displacement

vectors. Tracking contours and curves is particularly useful in the absence of any geometric model of the observed structures.

Deformable models can be used for visual tracking [24]. Such models can be used to represent the change in shape and position of a region of pixels. For each frame, the algorithm computes the model parameters that best explains the motion in the new frame. Deformable models are very useful in real-life situations involving conditions in the environment like human faces.

The method of *cross correlation* is usually used to search for a 3-D object's 2-D projection in different views of the same scene. It can also be used to get the relative displacement of the 2-D projection of a 3-D object from one image to the next. Cross correlation is used in [27] to detect pixel by pixel motion. Cross correlation methods are accurate but computationally expensive.

There are a number of algorithms in the literature for tracking point features. Jenkin [13] proposed a method for computing correspondences where he uses the *smoothness of velocity* of points. Sethi and Jain, [25], make use of this property over a series of monocular images. They exploit the fact that due to inertia, motion of an object cannot change abruptly. The trajectories of the feature points are computed so as to maximize a gain function. In this approach, however, the same set of features are required to be present in successive frames. The algorithm due to Sethi and Jain [25], is made use of in [7] to compute initial, partial trajectories which are subsequently used to initialize an algorithm based on the Stochastic Approximation method. Moving object trajectories can be found using the constraint of path and shape coherence constraint [10]. In [14] a proximal uniformity constraint is used to solve the correspondence problem. Initial matches are established using a gradient-based optical flow approach. A smoothness constraint similar to that of [25] is used in [29] where a smoothness measurement is computed for each trajectory and the best trajectory is selected using a heuristic approach. In the algorithm for point correspondences given in [32], intensity-based area correlation is used to match point features over consecutive frames. For feature localization, greater weightage is attached to points closer to the feature center. The method however, needs computations upto subpixel accuracy because of the camera motion compensation involved.

As in the computation of motion using the optical flow approach, the fact remains that tracking points is difficult in practice because of greater possibilities of errors due to noise and mismatches inasmuch as there are not too many distinguishing qualities, that can be found between points.

The tracking process can be made faster via appropriate hardware [1], [4]. Other algorithms which have been developed commercially such as Econolite Control Products, Inc., *'Autoscope vehicle detection system'*[*] have not been published yet. This has been used for real-time road intersection detection and control.

The algorithm presented in this paper selects *a priori* unknown features to be tracked over a sequence of images. However, changes in the brightness D.C. level in the pixels of the feature can be accommodated. Another important feature of the algorithm is the use of an adaptive feature matching threshold and the capability of detecting multi-pixel motion between frames. Further, any kind of motion can be tracked within the limits of number-of-pixels-moved-by-the-feature per frame. No subpixel computations are required. The method may be used as an alternative to optical flow for the analysis of motion. We also show how we used the feature tracking algorithm to extract the Focus of Expansion, using real image sequences.

1.1.2 Methods to extract Focus of Expansion(FOE)-An overview

Focus of Expansion, FOE, plays a vital role in the extraction of information about relative motion between the observer and 3-D objects. It is defined as the intersection of the imaging plane and the direction of motion [12], [5]. Most of the approaches available in the literature for FOE computation are optical-flow based. The results however, become extremely inaccurate in the presence of noisy data. Our approach, on the contrary, is based on feature tracking.

Until recently, FOE had been considered only in the context of the translatory motion. In Prazdny [23], it is shown that the rotation component can be canceled for two image points if the flow component along the line joining them is used. FOE, as used in [15], would be the intersection of the imaging surface and the axis of instantaneous translation, the surmise being that this would make it independent of the observer's instantaneous rotation. But the method is optical flow based and is bound to suffer from the effects of noise in real situations. In [21], a method for locating FOE is presented, based on computations in selected circular patches each of which is around an estimated FOE. The direction from the estimated to the true FOE is determined thereof. The intersection of these so-called FOE constraint lines for each patch gives the location of the FOE.

---

[*] Certain commercial equipment, instruments or materials are identified in this paper. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

A method that takes into account the uncertainty in determining the precise location of the FOE is given in [26]. The normal component of the optical flow due to translational egomotion is used in this approach. The method is insensitive to independent motion, if the portion of the scene corresponding to independently moving objects is small.

Another method, for the direct computation for the Focus of Expansion can be found in [12]. The approach involves the extremization of a function based only on the location of points in frames. However, the motion is restricted to translation only. Though correspondences need not be established, feature points however, need to be isolated. In [20], the scene is assumed to be rigid and the spatial gradient and the time of rate of change of brightness over the whole image is used.

In section 3, we show how the feature tracking algorithm can ben used to determine the FOE.

2. THE FEATURE TRACKING ALGORITHM

We present here an algorithm which tracks 2-D image features over several frames. The method needs no *a priori* knowledge about the motion or the scene being viewed. As far as selecting the features to be tracked is concerned, we do not use any pre-defined notion as to what constitutes a feature. In our algorithm, any n x n brightness pattern distinct from its neighbors, detected by the algorithm, can be used as a candidate feature.

The primary assumptions made in the development of the algorithm are:

1. Small inter-frame 2-D feature displacements, *i.e.*, features in the image do not move more than a few pixels per frame. In our implementation we allow up to 3 pixels per frame.
2. Sufficient texture in the image, *i.e.*, the image is rich enough in features. This assumption is made to ensure that features, as we define later on, exist.
3. The features remain in the Field-of-View (FOV) of the observer.
4. Slow brightness changes are allowed in the feature pixels.
5. Temporal continuity of the features, *i.e.*, the features exist in successive frames, within the limitations of assumption 4.

There are two stages in the algorithm. They are,

1. Feature selection.
2. Tracking of selected features.

2.1 FEATURE SELECTION

There are many algorithms for tracking point features. But tracking point features is disadvantageous because of practical difficulties like noise as well as greater possibilities of mismatches. Our algorithm tracks features which are chosen by the algorithm itself or in other words, which are trackable to the algorithm.

There are other methods to detect features like the Moravec interest operator, [18]. Features in images were picked using an interest operator, which returned regions of local maxima of a directional variance. Directional variance was measured over small windows. The sums of squares of differences between adjacent pixels in four directions, *i.e.*, horizontal,vertical and the two diagonals, were computed for each window and the window's interest measure was the taken as the least of these four sums. A feature was said to exist wherever the interest operator had local maxima.

We define a feature to be an *a priori* unknown n x n brightness pattern which is distinct enough to be tracked. By 'distinct enough', we mean that if the n x n feature candidate location were to be shifted as shown in Figure 2.1, in its immediate neighborhood, it would have very little correlation between itself and the new n x n neighboring brightness patterns (By feature location we refer to the top-left corner pixel of the feature). In other words, a feature can be defined as a sharp peak on a matching surface which is constructed using neighboring n x n patterns and the feature itself.

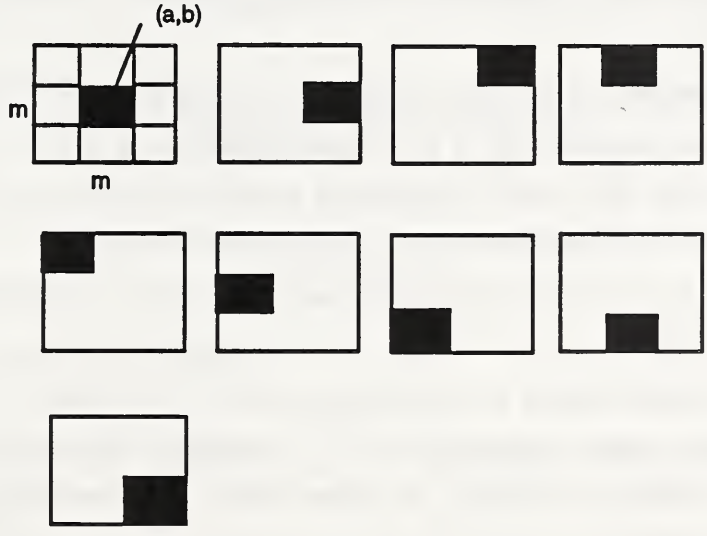


Figure 2.1 : Each m x m square represents the search matrix. The shaded portion represents the location of the top-left corner of the feature candidate.

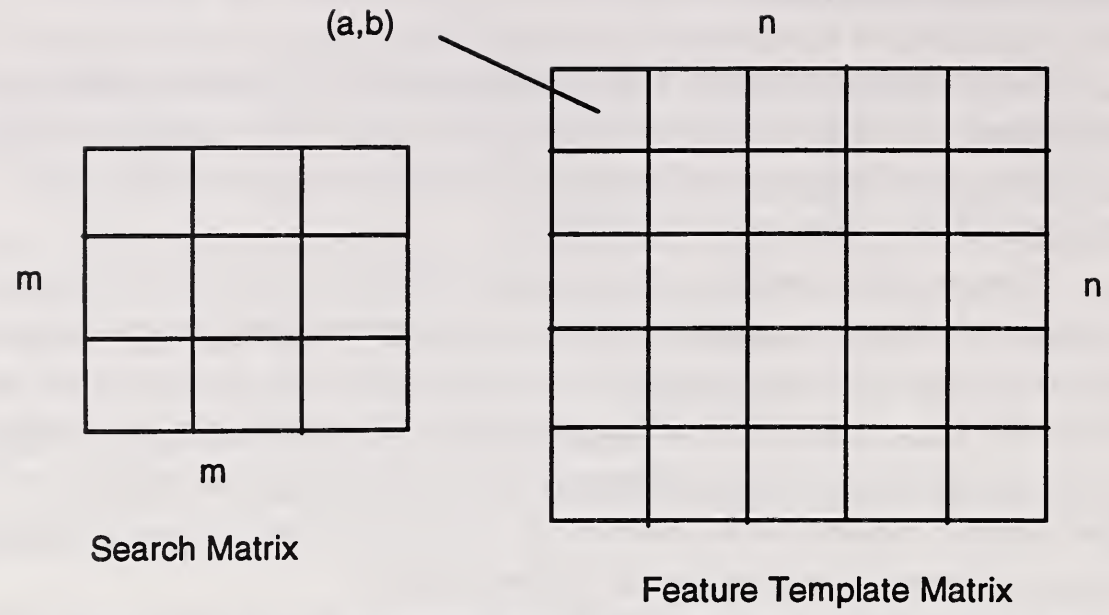


Figure 2.2 : A representation of the Search Matrix and the Feature Template Matrix.

The following procedure is implemented to select a feature:

Starting at any pixel located at (a,b) in the first image of the sequence to be analyzed, we scan the image for a feature candidate. Once a feature is found we may continue to search for the next feature. We make sure that the pixel location (a,b) is not located at the edges of the picture because that would prevent our doing the local search as described below.

1) Pixel (a,b) is assumed to be the top-left pixel of the feature. Keeping (a,b) as the starting point we construct an n x n feature matrix, say a 5 x 5 matrix, of pixel intensities. We call this matrix our *feature template matrix* '**ms**' (Figure 2.2).This *feature template matrix* contains image intensities at that location.
2) We construct an m x m *local search matrix*, say 3 x 3 with (a,b) being the center of this matrix (Figure 2.2).
3) The *feature template matrix* is translated up, down, left, right and across both the diagonals of the search matrix, *i.e.*, $m^2 - 1$ locations. At each such location, we compute the matching coefficient as given below. The associated parameters are designated as follows:

ms a feature template of size n x n,
ms[i][j] an element of **ms**,
im a sub-image of size n x n,

net the sum of differences between **ms** and **im**,

ave the average D.C. brightness level change per pixel,

n_1 the variance of the difference between **ms** and **im** taking into account D.C. brightness level,

n_2 the sum of squared differences between the pixel values of **ms** and **im**,

d the normalizing factor

c_1 the mismatch coefficient contribution without change of brightness D.C. level,

c_2 the mismatch coefficient contribution with change of brightness D.C. level,

c the net mismatch coefficient,

cf the matching coefficient.

Then,

$$net = \sum_{i=1}^{n}\sum_{j=1}^{n} (ms[i][j] - im[i][j]) \tag{1}$$

$$ave = net / (n^2) \tag{2}$$

$$n_1 = \sum_{i=1}^{n}\sum_{j=1}^{n} (ms[i][j] - im[i][j] - ave)^2 \tag{3}$$

$$n_2 = \sum_{i=1}^{n}\sum_{j=1}^{n} (ms[i][j] - im[i][j])^2 \tag{4}$$

$$d = \sum_{i=1}^{n}\sum_{j=1}^{n} (ms[i][j] + im[i][j])^2 \tag{5}$$

$$c_1 = (n_1 / d)^2 \tag{6}$$

$$c_2 = (n_2 / d)^2 \tag{7}$$

$$c = c_1 * k_1 + c_2 * k_2 \tag{8}$$

where $k_2 = 1 - k_1$ and $0 \le k_1 \le 1$, $0 \le k_2 \le 1$;

$$cf = (1 - c) \tag{9}$$

k_1 and k_2 are used so that we do not totally ignore the contribution of the self brightness changes.

Using these matching coefficients, an m x m matching surface is constructed keeping (a,b) as the center location of this surface. An n x n pattern is called a feature if all the gradients between the peak on the matching surface and its neighbors are greater than

a threshold value, that we specify *a priori*. In other words (a,b) is a peak on the matching surface and is our feature location.

The matching coefficient computed at our feature location (a,b) then becomes our matching threshold for the second frame of the sequence.

Table 1 gives the number of computations involved for an n x n feature template matrix.

This whole procedure of feature selection needs to be done only once, in the first frame in the sequence of frames to be analyzed.

Additions	Subtractions	Multiplications	Divisions
$8n^2+1$	$3n^2+2$	$15n^2+8$	3

Table 1 : Number of computations per feature at a search matrix location.

2.2 SUBSEQUENT TRACKING

Once the features to be tracked have been selected in the first frame k=1, subsequent tracking is done as given below.

2.2.1 Tracking in the second frame of the sequence

During the second frame, frame number k = 2, we center the search matrix of size p x p, p ≥ m, (as in Figure 2.2), around the location of our feature, the idea being that using a larger search matrix would help to establish the initial motion pattern. Once a pattern has been established, the search matrix can be changed accordingly.

The algorithm performs a local search to lock on to the feature. The location of the maximum matching coefficient gives the new location of the feature in the second frame. The equations given in section 2.1 are used in computing the matching coefficient.

2.2.2 Subsequent Frames

For k ≥3, there are 2 stages involved in the tracking:

1) Local spatial search

2) Check for directional temporal continuity

The algorithm centers the search matrix, this time of a size smaller than or equal to that used in the second frame of the sequence, around the location of the feature in the previous frame.

Once this local search has been done, all matching coefficients above an *a priori* specified threshold, are further examined in the following manner.

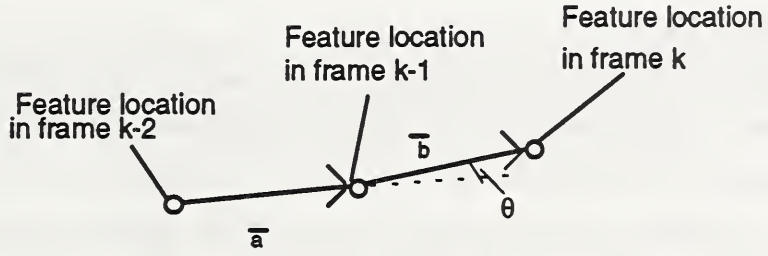
The idea is to check if the direction of motion of the feature from frame k-2 to k-1 is similar to that between frame k-1 and the current frame k. Refer to Figure 2.3.

For each candidate location obtained using the spatial search, the best match is the one that minimizes the constraint 'β' given in equation (11).

$$\cos\theta = \frac{\bar{a}.\bar{b}}{|\bar{a}||\bar{b}|} \tag{10}$$

where $\bar{a}$ and $\bar{b}$ are vectors as shown in Figure 4.

$$\beta = 1 - \cos\theta \tag{11}$$



$\bar{a}$ is the motion vector between frames k-2 and k-1

$\bar{b}$ is the motion vector between frames k-1 and k

o represents feature location as pixel coordinates

Figure 2.3 : Representation of the check for motion vector continuity.

2.3 FEATURE AND THRESHOLD ADAPTATION

If the feature located is a very good match to the feature in the earlier frame, the pixel intensities in the n x n feature are updated to that of the new location. The matching threshold is also updated under such circumstances. This will ensure that the original 2-D feature can still be detected even if it undergoes changes. This will avoid the constraint that the same set of feature points have to be present in successive frames, as is necessary

in [25]. There are many possibilities as to how the threshold can be adapted. Given below is the manner in which we have implemented the threshold adaptation step.

In the threshold adaptation process, we try to take into account the fact that brightness changes in the feature may also mean brightness changes in its neighbors. This should add to the robustness of the tracking.

Figure2.4 shows the matching coefficients used to determine the threshold.

a1	a2	a3
a4	a5	a6
a7	a8	a9

Figure 2.4 : Grid of coefficients used to determine the Feature Matching Threshold. a1..a9 are the matching coefficients obtained using the equations given in Section 2.1.

There are two cases to be considered:

1. **A not-very-distinct feature:**

By 'a not-very-distinct feature' we refer to a low peak on the correlation surface. In other words, the feature is not as distinct from its neighbors as it was at the beginning of the image sequence. The idea in this case is that if a feature is not very distinct from its neighbors, it is easier to track it taking into account the presence of its neighbors. In this case, we take the mean of a2, a4, a6, a8 and a5 (Figure 2.4).

$$\text{Threshold} = (a2+a4+a6+a8+a5)/5 \tag{12}$$

2. **A very distinct feature:**

By 'a very distinct feature' we mean a sharp peak on the matching surface.

In this case, the matching coefficient is used as the threshold.

$$\text{Threshold} = a5 \tag{13}$$

A threshold to determine the sharpness of the peak is specified *a priori.*

A limited tolerance of matching threshold is allowed while determining possible matches during the tracking.

Note : Once the threshold has been adapted in a particular frame, the matching coefficient in the next frame is computed as follows.

If equation (12) is used while computing the threshold, then the matching coefficient is computed in the same manner as given in equation (14)

$$cf = (a2+a4+a6+a8+a5)/5 \tag{14}$$

where the coefficients are arranged as shown in Figure 2.4.
Otherwise,

$$cf = a5 \tag{15}$$

This is done to ensure that both the matching coefficient and the threshold can be compared, by computing them in similar fashion.

2.4 OCCLUSIONS

In situations where the feature is lost or partially occluded for a short time period, the search window is simply restored to its original size, as used in the second frame and the search for the best match is done as it was done in the second frame. The decision as to whether the algorithm is momentarily lost or whether there is occlusion, can be inferred from the peak of the matching surface. A very low peak would indicate occlusion while a peak that is simply lower than the matching threshold would indicate that the algorithm is momentarily lost.

3. EXPERIMENTS

Through our experiments, we aimed at obtaining two types of results.
a) Results of the tracking for both translational and rotational motion.
b) To use the feature tracking algorithm to extract the FOE.

Experimental Set-up - The implementation was done using a camera mounted on a six-degree-of-freedom vision-based flight simulator robot controlled by an 80486-based PC. The simulator has been obtained by modifying an IBM Clean Room Gantry Robot (Figure 3.1). The modification is such that the robot's main controller is not being used. Instead it is controlled by the PC that supplies analog signals to the six control loops, each corresponding to a velocity input.

Experiments were performed with the unstructured environment shown in Figure 3.2. To determine the real FOE, a '+' was marked on the TV monitor at some location. A similar '+' was also placed in the 3-D environment within the FOV of the camera. The camera was translated such that the '+' marked on the TV monitor and the '+' in the 3-D environment as seen by the camera coincided at each camera location during the translation. The location of the '+' thus gave us the location of the real FOE.

Experiments- In our experiments, the Focus of Expansion was defined as the intersection of the image plane and the direction of motion.

A series of 120 frames were acquired in succession using a frame grabber. The algorithm tracked the features selected, in each frame of the sequence. The feature points selected were all at different distances from the camera. In our experiments, the size of the features used were 5 x 5 while the search regions used were 3x 3 and 5 x 5. The tolerance used in the threshold value for determining feature updating was 3% of the threshold. The tolerance in the threshold used to determine matches during tracking was 10%. Occlusion thresholds used were 50% of the threshold value. k_1 and k_2 used in the computation of the matching coefficient were 0.8 and 0.2, respectively.

The trajectories of the features were plotted over a sequence of frames. Then using a least squares approximation a function was plotted to obtain each feature's smoothened trajectory in the image sequence. The lines thus plotted for each of the features were extended to find their point of intersection. This gave us the location of the FOE.

Results - Experimental results are presented in Figure 3.3-3.13.

In this section, the results presented are as follows:

1) Feature trajectories obtained using the feature tracking algorithm.
2) Using the feature trajectories to extract the FOE using real image sequences.

Figures 3.3-3.10 represent trajectories and the extraction of the FOE obtained by tracking features in different image sequences. In all these experiments, motion was confined to pure translation along the camera optical axis. Figures 3.11-3.13 represent trajectories for different motions. The location of the FOE found using the tracking algorithm was found to be in the vicinity of less than 4 pixels from the true FOE.

4. DISCUSSION

Presented here is a simple feature tracking algorithm, which has been used to extract the Focus of Expansion in a real image sequence. The definition of FOE that we have used is similar to the definition found in [5], *i.e.*, we define the FOE to be the point of intersection of the image plane and the direction of motion. The method was found to be quite robust even under noisy conditions.

Our contributions are as follows:

1. The algorithm presented here tracks unknown 2-D features obtained from an unknown 3-D environment. The features can undergo temporal changes in brightness.
2. This is a pixel-based method. Hence it can work in parallel.
3. No subpixel calculations are necessary.
4. The algorithm can track multiple 2-D features on multiple 3-D objects.

5. The method can tolerate more than one pixel motion between frames (in our experiments we track up to 3 pixels motion between frames)
6. It can be used for many applications such as extracting the Focus of Expansion (FOE).
7. There is no need to compute optical flow. In fact the method could be used as an alternative to optical flow methods.
8. Since the algorithm initiates its own feature search, the method can be applied in indoor as well as outdoor environments.
9. The algorithm can tolerate any kind of motion, *i.e.*, translation and rotation.

Like all template matching algorithms, this method faces the problem of numb[illegible] computations. Since, the initial search, *i.e.*, in the second frame, is not based on any kind of *a priori* knowledge about the 3-D environment, it is vital that the initial matches are correct, as this affects subsequent tracking. The algorithm also faces problems in the face of occlusions, but this can be overcome using the temporal continuity constraint. The feature selection process, the most time-consuming, can be modified or done away with if more *a priori* information is available.

The location of the FOE extracted using the tracking algorithm can be made more precise by using a larger number of features.

5. FUTURE WORK

The proposed algorithm can be used for object segmentation in a situation involving multiple clustered objects. More work is necessary to make it computationally more efficient. The problem of occlusions can be viewed in greater depth. In case the tracking algorithm gets lost due to occlusions, the trajectories obtained using the above algorithm may not yield proper results. In such situations, the image velocities from previous frames can be used for correct tracking.

Another way to overcome the problem of occlusion will be to make use of the motion continuity constraint to determine the feature location during periods of occlusion.

It may also be possible to build hardware to implement the algorithm in future. Attempts are being made to incorporate more *a priori* data to make the method more robust. One approach would be to incorporate the 6 parameter camera constants to improve the correlation.

ACKNOWLEGEMENTS

The authors would like to thank Dr. Neelakantaswamy, Hakan H Yakali, Murali Gopinathan and Sridhar Kundur for their suggestions made during the preparation of this paper.

REFERENCES

[1] Albus, J.S. and Hong T.H., "Motion, Depth and Image Flow", *IEEE Proc. CVPR* , 1990, pp. 1161-1170.

[2] Ayache N., Cohen, I. and Sulger, P., "Tracking Points on Deformable Objects Using Curvature Information", P*roc. ECCV,* 1992, pp. 458-466.

[3] Broida, T.J., Chellappa, R. and Chandrashekar, S., "Recursive 3-D Motion Estimation From a Monocular Image Sequence", I*EEE Transactions on Aerospace and Electronic Systems*, Vol 26, No 4, July 1990, pp. 639-655.

[4] Burt, P.J., Bergen, J.R. *et al* , "Object Tracking with a Moving Camera", *IEEE Proc. CVPR*, 1989, pp. 2-12.

[5] Cutting, 1986, **Perception with an Eye For Motion,** MIT Press.

[6] Deriche, R. and Faugeras, O., "Tracking Line Segments", I*mage and Vision Computing*, Vol 8 , No 4, Nov 1990, pp. 261-270.

[7] Fletcher, M.J., Warwick, K., Mitchell, R.J., "The Application of a Hybrid Tracking Algorithm to Motion Analysis", *IEEE Proc. CVPR*, 1991, pp. 84-89.

[8] Gennery, D., "Tracking Known 3-D Objects", *Proc AAAI 2nd Natl. Conf. Artif. Intell.*, Pittsburgh, PA, 1982, pp. 13-17.

[9] Gennery, D., "Visual Tracking of Known 3-D Objects", I*ntl. Journal of Computer Visio*n, 7:3, 1992, pp.243-270.

[10] Huang, C. and Wu, C. ,"Dynamic Scene Analysis Using Path and Shape Coherence", *Pattern* R*ecognition*, Vol 25, No 5, 1992, pp. 445-461.

[11] Jain, J.R. and Jain, A.K., "Displacement Measurement and Its Application inInterframe Image Coding", *IEEE Trans.*, COM -12, 1981, pp. 1799-1808.

[12] Jain, R.,"Direct Computation of The Focus of Expansion", *IEEE Transactions on PAMI*, Vol PAMI-5, No 1, Jan 1983, pp. 58-65.

[13] Jenkin, M., "Tracking Three Dimensional Moving Light Displays", P*roc. Workshop Motion*: R*epresentation Contr.*, toronto, Ont., Canada, 1983, pp. 66-70.

[14] Krishnan, R. and Shah, M., "Establishing Motion Correspondence", *IEEE Proc. CVPR,* 1991, pp. 103-108.

[15] Lobo, N.D.V. and Tsotsos, J.K., "Using Collinear Points to Compute Egomotion and Detect Nonrigidity", *IEEE Proc. CVPR,* 1991, pp. 344-351.

[16] Meyer, F. and Bouthemy, P., "Region-based Tracking in an Image Sequence", P*roc ECCV,* 1992, pp. 476-484.

[17] Micheli, E.D., Torre, V. and Uras, S., "The Accuracy of The Computation of Optical Flow and of The Recovery of Motion Parameters", *IEEE Trans. PAMI*, Vol 15, No 5, May 1993, pp. 434-447.

[18] Moravec, H.P., "The Stanford Cart and The CMU Rover", P*roceedings of The IEEE*, Vol 71, No 7, July 1983, pp. 872-884.

[19] Naonari and Kenji, "Tracking Moving Contours Using Energy-minimizing Elastic Contour Models, P*roc ECCV* 1992, pp. 453-457.

[20] Negahdripour, S. and Horn, B.K., "A Direct Method For Locating The FOE", C*VGIP*, 46, 1989, pp. 303-326.

[21] Negahdripour, S. and Ganesan, V., "Simple Direct Computation of The FOE with Confidence Measures", *IEEE Proc. CVPR* , 1992, pp. 228-235.

[22] Ogata, M. and Sato, T., "Motion-detection Model with Two Stages : Spatio-temporal Filtering and Feature Matching", *J. of Opt Soc of America*, Vol. 9, No. 3, Mar 1992, pp. 377-387.

[23] Prazdny, K., "on The Information in Optical Flows", *CVGIP*, Vol. 22(2), 1983, pp. 239-259.

[24] Rehg, J. and Witkin, A., "Visual Tracking with Deformation Models", P*roc. 1991 IEEE Intl. Conf. on Robot. and Automation*, April 1991, pp. 844-849.

[25] Sethi, I.K. and Jain, R., "Finding Trajectories of Feature Points in a Monocular Image Sequence", *IEEE Transactions on PAMI*, Vol PAMI-9, No 1, Jan 1987, pp. 56-73.

[26] Sharma, R. and Aloimonos, J., "Robust Detection of Independent Motion: An Active and Purposive Solution", *CAR-TR-534*, January 1991.

[27] Spoer, P., "Displacement Estimation For Objects on Moving Background", in Huang T S(Ed): I*mage Sequence Processing and Dynamic Scene Analysis*, Springer-Verlag, Berlin, 1983, pp. 424-436.

[28] Vega-Riveros, J. and Jabour, K., *IEE Proceedings*, Vol 136, Pt 1, No 6, Dec 1989, pp. 397- 404.

[29] Vincent, S.S. Hwang, "Tracking Feature Points in Time-Varying Images using an Opportunustic Selection Approach", *Pattern Recognition*, Vol 22, No 3, pp. 247-256.

[30] Yen, B.L. and Huang, T.S., "Determining 3-D Motion/structure of a Rigid Body Using Straight Line Correspondences", in T S Huang (Ed) : *Image Sequence Processing and Dynamic Scene Analysis,* Springer-Verlag, Berlin, 1983, pp. 365-394.

[31] Zhang, H. and Zhenya, H., "A New Motion Estimation Algorithm", I*EEE Pacific Rim Conf. on Comm., Computers and Signal Processing*, May 1991, pp. 619-622.

[32] Zheng, Q. and Chellappa, R., "Point Correspondence and Motion Detection in Long Image Sequence", *IEEE Proc. CVPR*, 1993, pp. 661-671.

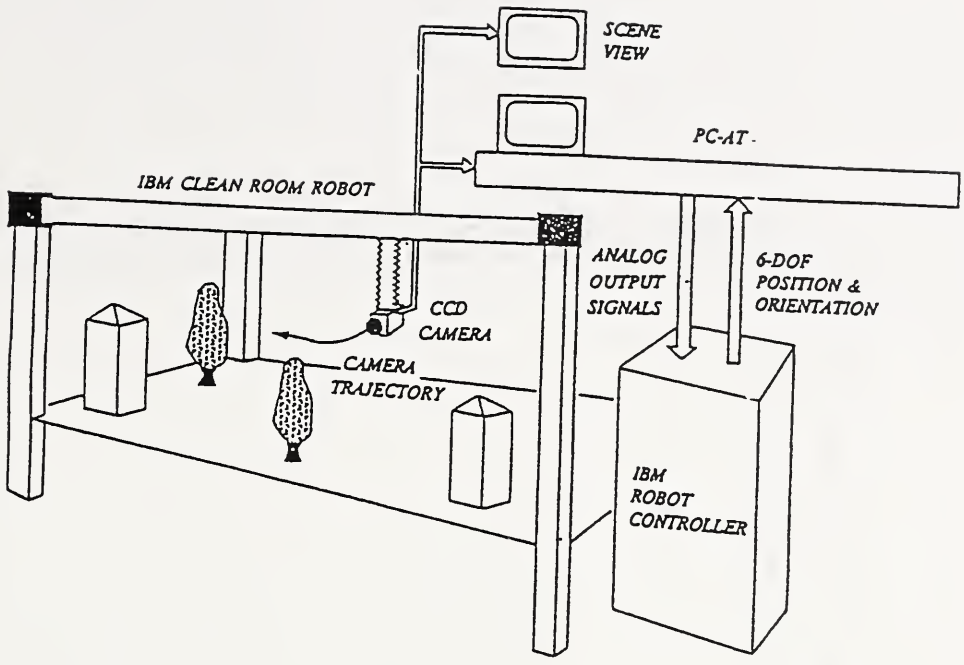


Figure 3.1 : IBM Robot used in experiments.

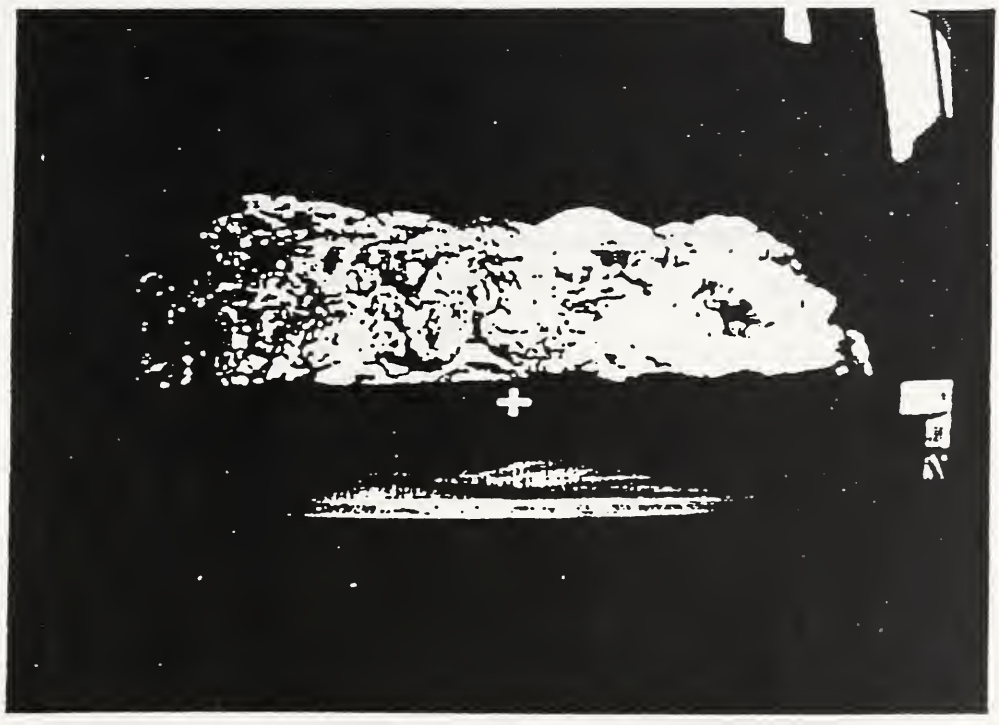


Figure 3.2 : Environment used for Feature Tracking and FOE experiments.

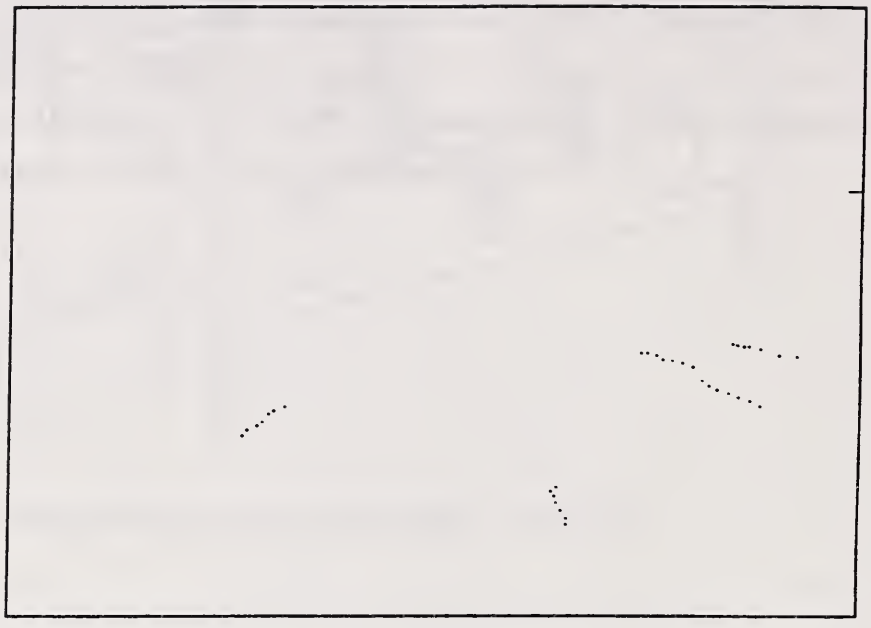


Figure 3.3 : Trajectories for 5 features, translatory motion.

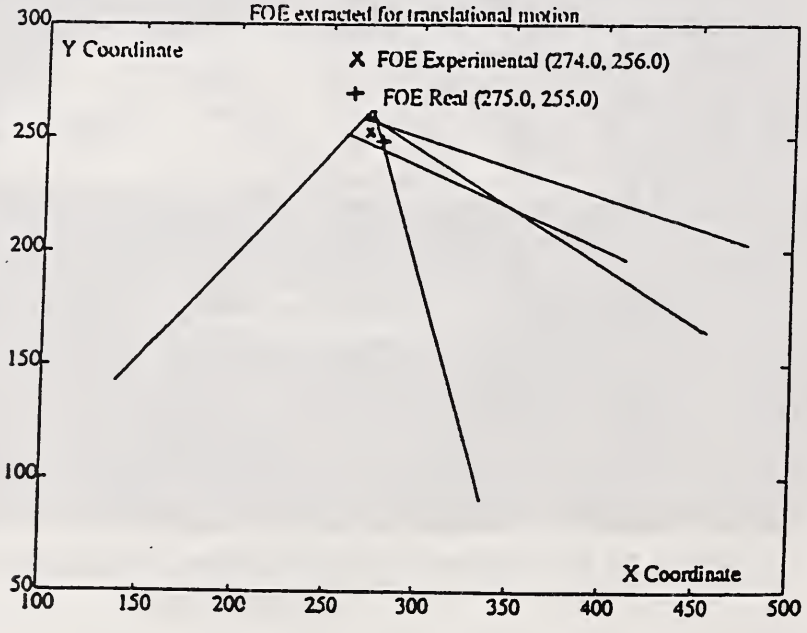


Figure 3.4 : FOE extracted for trajectories in Figure 3.3.

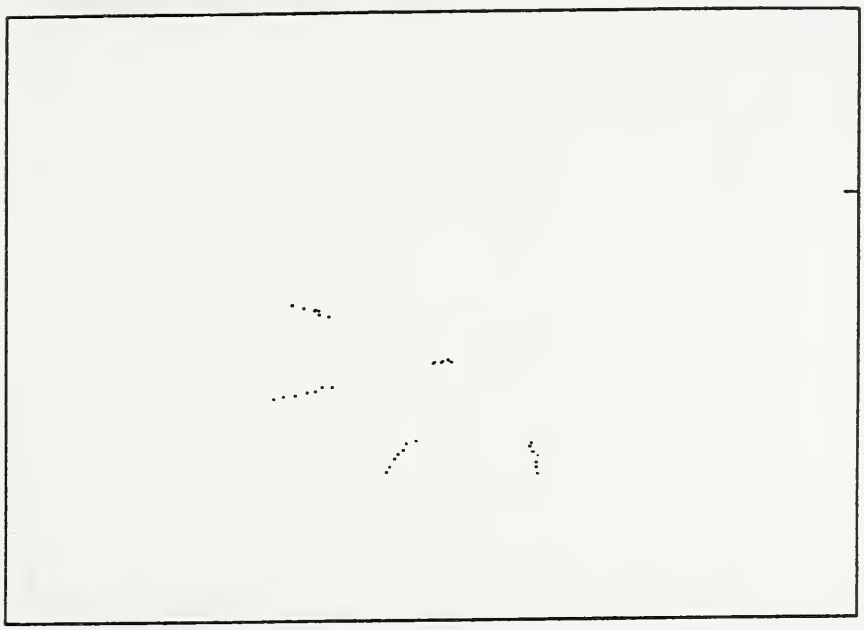


Figure 3.5 : Trajectories for 5 features, translatory motion.

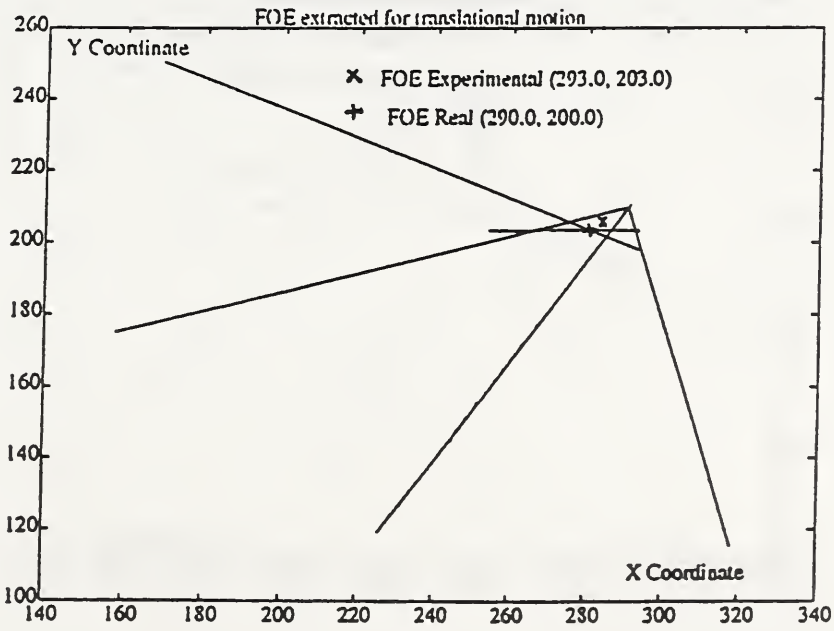


Figure 3.6 : FOE extracted for trajectories in Figure 3.5.

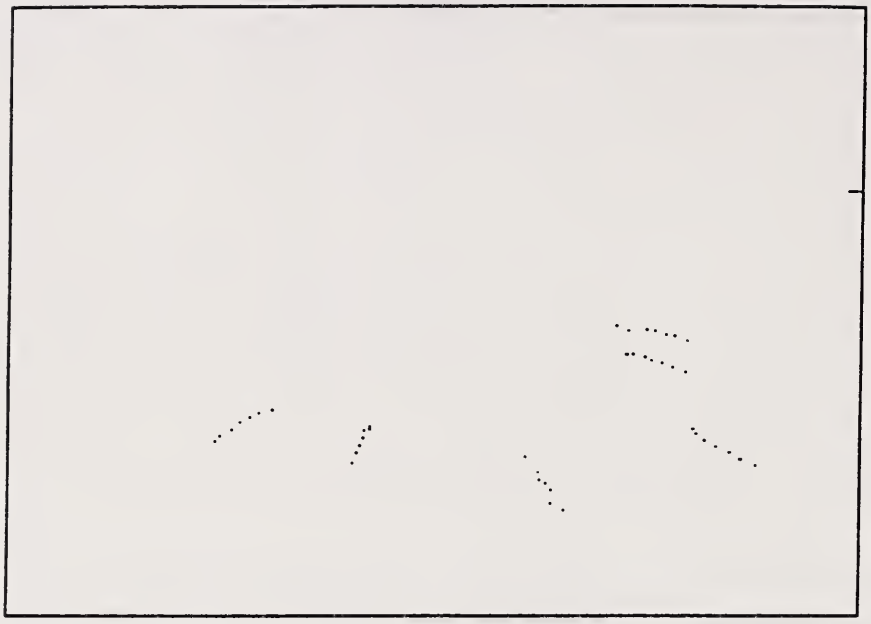


Figure 3.7 : Trajectories for 6 features, translatory motion.

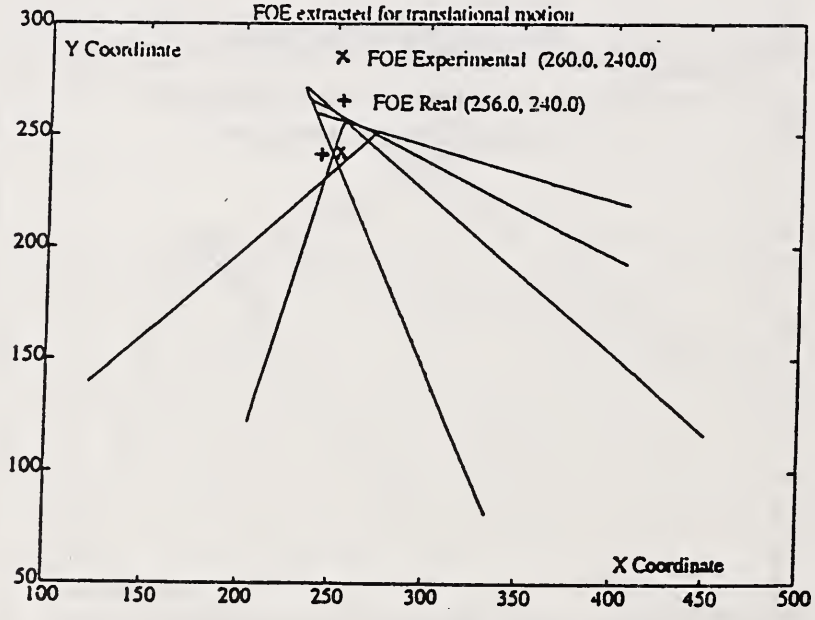


Figure 3.8 : FOE extracted for trajectories in Figure 3.7.

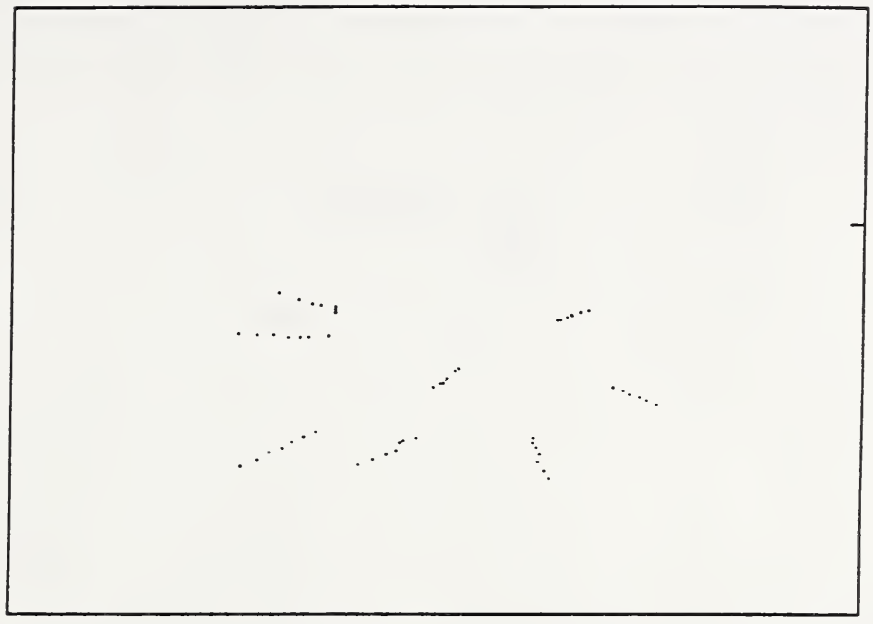


Figure 3.9 : Trajectories plotted for 8 features

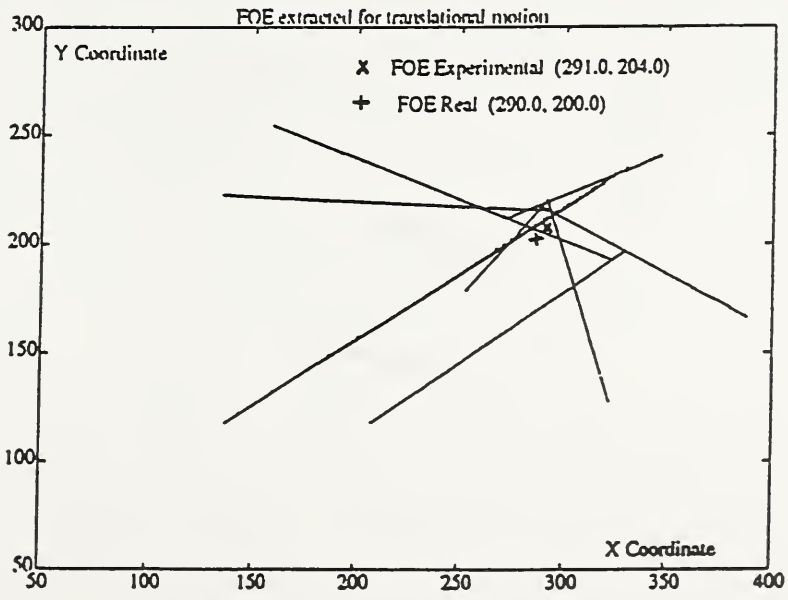


Figure 3.10 : FOE extracted for trajectories in Figure 3.9.

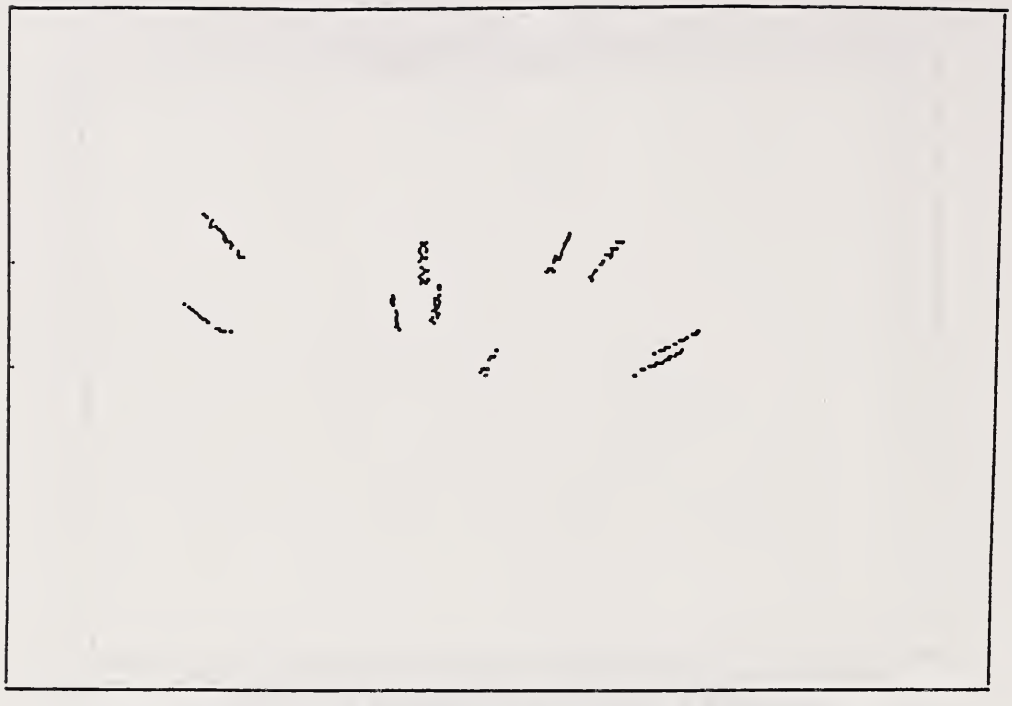


Figure 3.11 : Trajectories for 10 features, translatory motion.

Figure 3.12 : Trajectories for 2 features, rotational and translational motion.

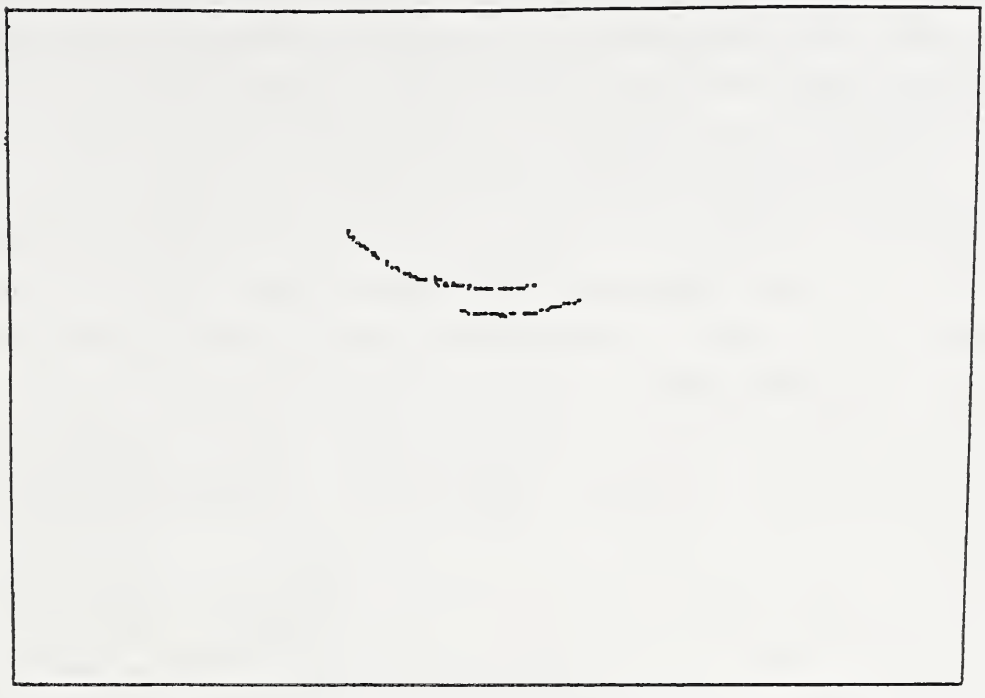


Figure 3.13 : Trajectories for 3 features, rotational and translational motion.

Abstract

This paper describes a robust, adaptive algorithm for tracking image features, both spatially and temporally, over a sequence of monocular images. The algorithm assumes no *a priori* knowledge about the features to be tracked, or the motion of the camera/the objects, in the 3-D scene. The features to be tracked are selected by the algorithm and these are the peaks of a *'matching surface'* constructed from the first image of the sequence to be analyzed. *Any kind of motion* can be tolerated keeping in mind the pixels- per-frame motion limitations. In our experiments, we have used the algorithm for tracking up to 3 pixels motion between frames. Absolutely *no subpixel calculations* are necessary. Taking into account constraints of temporal continuity, the algorithm uses simple and efficient predictive tracking over multiple frames. The algorithm accepts a slow, continuous change of brightness D.C. level in the pixels of the feature. Trajectories of features on multiple objects can be computed. As an application of the algorithm we show how the algorithm can be used to find the Focus of Expansion (FOE) using a sequence of real images.